Certifying Compilation with Secure Virtual Architecture

Vikram Adve

Associate Professor

University of Illinois at Urbana-Champaign

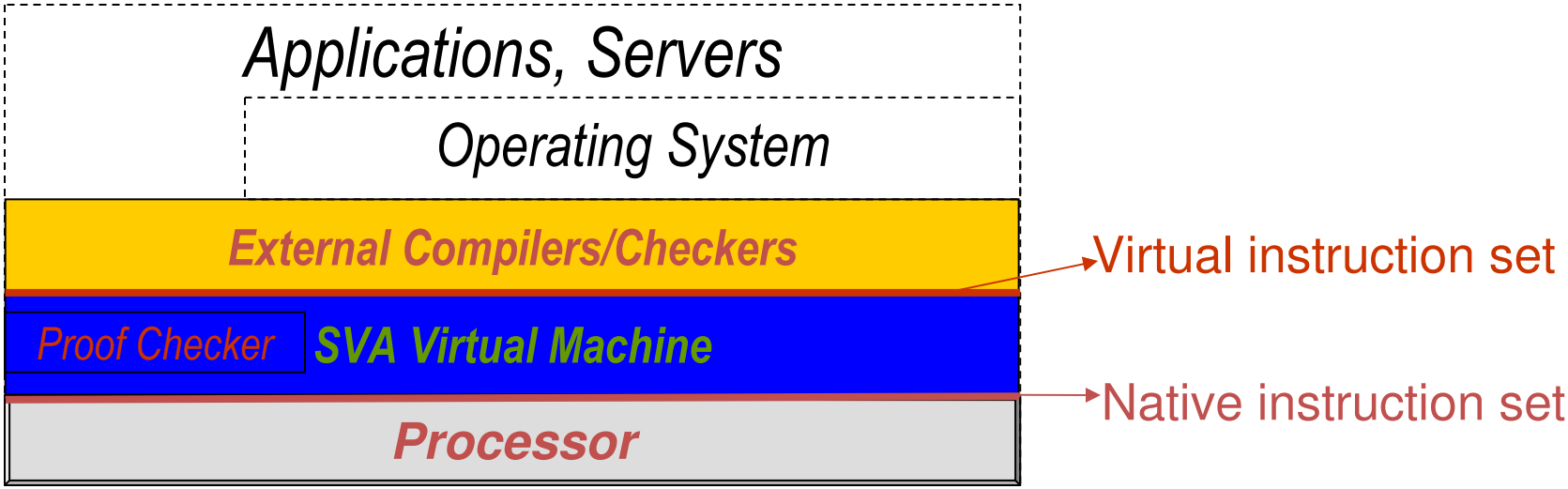
vadve@illinois.edu

(217) 244-2016

http://llvm.org/~vadve

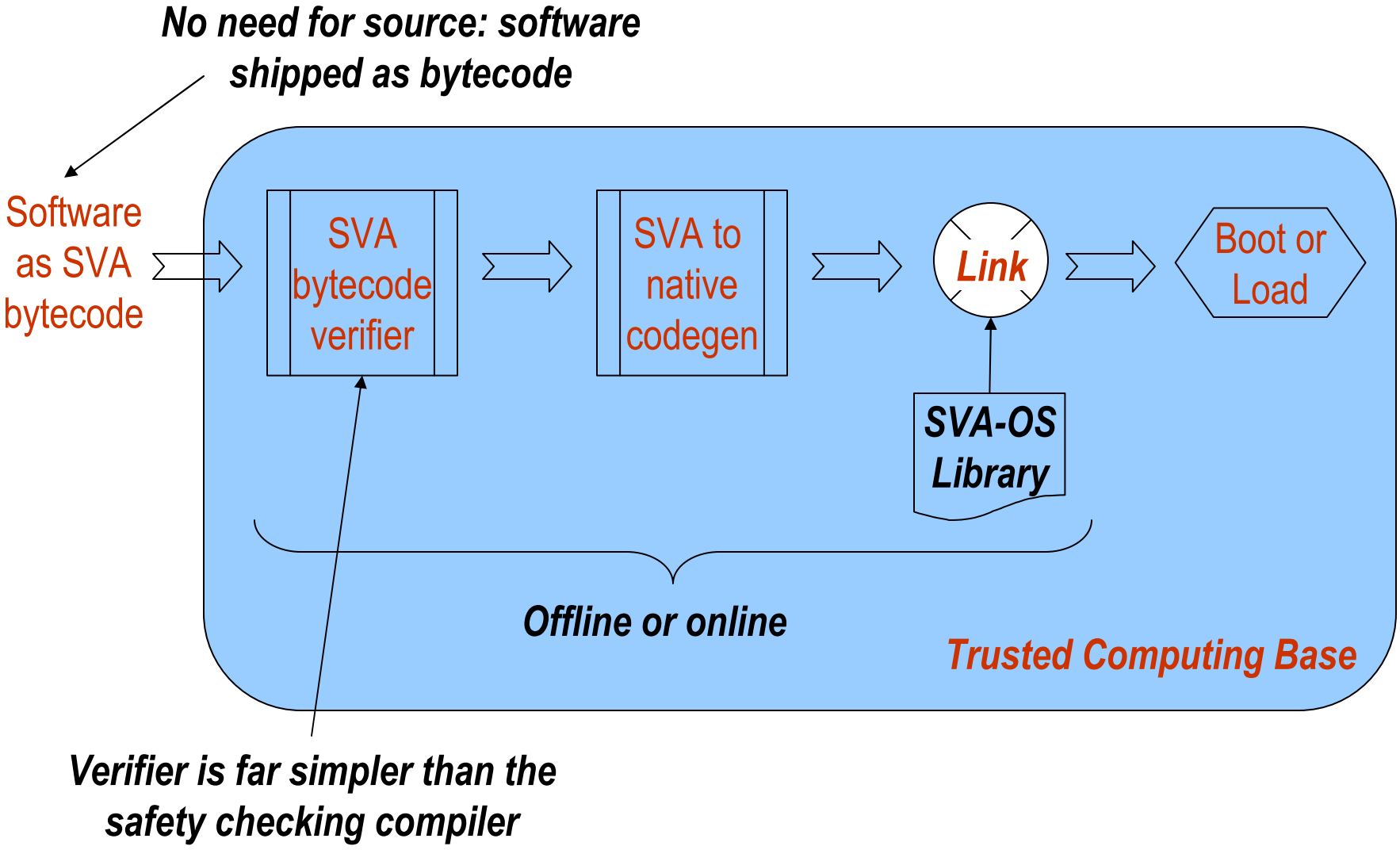
Secure Virtual Architecture

- *Compiler-based VM below operating system*
- Enforces safety properties via analysis and transformation
- Supports type-unsafe languages (e.g., C/C++)
- Supports application/kernel code (e.g., Linux)



Criswell et al., [SOSP 2007]

Certifying Compilation



Advantages of our Approach

Building blocks for a complete system solution

- Virtual instruction set
  - Comprehensive software representation (source code unnecessary)
  - Special instructions make OS, system calls easier to analyze
  - Sophisticated analysis capabilities
- Single, unified certification strategy for all software
- Ready to go:
  - Linux 2.4 ported; Linux 2.6 port underway
  - Memory safety for applications *and* entire Linux kernel

[*PLDI 2006, SOSP 2007, Usenix Sec. 2009*]

Seeking Capabilities In…

- Formal Methods
  - Formal framework for security certification
- Static analysis for security properties
  - Can operate on SVA bytecode (no source needed)
  - Orthogonal to run-time fault detection, isolation in SVA
- Browser Security
  - Web apps an important class of application
  - Many browser vulnerabilities *orthogonal* to system software bugs
  - Browser is *model* of extensible, open platforms